グラフオートマトンによる部分グラフ探索アルゴリズムの開発と実装 (理論計算機科学の最先端)

AUTHOR(S):

藤芳, 明生

CITATION:

藤芳, 明生. グラフオートマトンによる部分グラフ探索アルゴリズムの開発と実装 (理論計算機科学の最先端). 数理解析研究所講究録 2017, 2040: 1-9

ISSUE DATE:

2017-07

URL:

http://hdl.handle.net/2433/236900

数理解析研究所講究録
第2040巻 2017年 1-9

グラフオートマトンによる 部分グラフ探索アルゴリズムの開発と実装

藤芳明生
茨城大学工学部情報工学科
akio.fujiyoshi.cs@vc.ibaraki.ac.jp

Akio Fujiyoshi
Department of Computer and Information Sciences,
Ibaraki University

1 はじめに

本発表は、新しい部分グラフ探索アルゴリズムを提案するものである。提案アルゴリズムは、部分グラフ同型性判定問題を解けるだけでなく、グラフオートマトンによって受理される部分グラフの中で最適な物を探索することもできるようになっている。用いられるグラフオートマトンも新しいものであり、木オートマトンに非常にシンプルな拡張を行うことで得られたものである。

部分グラフ同型性判定問題とは、2つのグラフ G と H が与えられたとき、G が H を部分グラフとして含むかどうかを判定する問題である。H が G と同じ頂点数の単純サイクルならば、この問題は、G のハミルトン閉路問題と同じことなるため、この問題もNP完全問題である。他にも、ハミルトンパス問題やクリーク問題などの多くのNP完全問題を含んでいる。部分グラフ同型性判定は、化学物質の部分構造検索をするために実際に必要とされている。そのため、部分グラフ同型性判定はNP完全問題であるが、実用上十分に高速な実装が必要とされてきた。

既存の部分グラフ同型性判定のアルゴリズムについて述べる。G と H の頂点の対応をすべて調べるようなナイーブなアルゴリズムならば、$O(|V(G)|^{|V(H)|})$ 時間で解くことができる。高速に動作する有名なアルゴリズムには、Ullmann アルゴリズム [1, 2] や、VF2 アルゴリズム [3] 等が存在し、これらのアルゴリズムの実装も公開されている。Ullmann アルゴリズムと VF2 アルゴリズムは、最悪の実行時間は階乗時間とされているが、平均的な入力については結構高速に動作することが知られている。また、最大次数及び Tree-Width が制限されているならば、Matoušek と Thomas による多項式時間のアルゴリズム [4] も存在する。

では、なぜ新しい部分グラフ探索アルゴリズムを開発したいと思ったのかについて述べる。それは、部分グラフ同形性判定をするだけでは不十分であり、次に示すような拡張問題も解きたいと思ったからである。

- グラフ H の代わりにグラフオートマトンを入力とし、条件を満たす（グラフオートマトンに受理される）部分グラフを探索したい。
- 条件を満たすサイズ（または重み）が最大/最小の部分グラフを探索したい。
- 探索範囲を、全域部分グラフまたは誘導部分グラフに絞って探索したい。
- 単純グラフだけでなく、ラベル付き有向多重グラフの部分グラフを探索したい。

提案アルゴリズムは、このすべてに対応している。

この拡張により、提案アルゴリズムは様々な最適化・探索問題が解けるようになった。パスや閉路を受理するグラフオートマトンは容易に構成できるので、最長パス問題や、最長閉路問題を解くことはすぐに実現できる。他にも、閉路を受理するグラフオートマトンで、「全域部分グラフ探索」かつ「重み最小探索」のモードで動かせば、巡廻セールスマン問題が解けるようになる。$K_{3,3}$ 及び K_5 の細分を受理するグラフオートマトンで、「任意の部分グラフ探索」のモードで動かせば、グラフの非平面性判定ができる。

2 新しい部分グラフ探索アルゴリズムの開発

新しい部分グラフ探索アルゴリズムの開発に辺り、正規表現による文書検索と同じようなことをグラフでもできるようにすることや、将来的には、機械学習などに利用することも考慮に入れることとした。また、実装するときに、化学物質の構造を表現するグラフ（化学グラフ）に対しては、特に高速に動作できることを念頭に置いた。

化学グラフとは、化学物質の構造を表現した化学構造式をグラフと見立てた物である。頂点は原子の種類でラベル付けされており、辺は結合の種類によってラベル付けされている。原子が他の原子と結合できる数には制限があるため、最大次数が定数で制限されたグラフであると言える。また、ほとんどの化学グラフは平面的であることが知られている。さらに、医薬品の候補化合物に限れば、Tree-Width は小さいことが知られている。実際、ChEMBL に登録されている 635,933 個の医薬品及び医薬品候補化合物の Tree-Width を調べてみたところ、Tree-Width が 4 以上の物は 343 個 (0.05%) しか無いことが分かった。

2.1 アルゴリズムの開発方針

アルゴリズムの開発方針として、化学グラフを主な対象とするため、次数および Tree-Width が小さいグラフに対し、高速に動作するように設計することとした。

発表者は、「木オートマトンと Tree-Width=2 のグラフを入力とし、木オートマトンが受理する全域木を探索するアルゴリズム」は既に実装済みであり [5, 6]、これを種に開発することにした。これには、以下のような拡張が必要となる。

1. 木オートマトンを入力としていた部分を、グラフオートマトンを入力とするように拡張する。
2. 部分グラフの探索範囲を全域部分グラフに制限していた部分を、任意の部分グラフを探索できるように、更に、任意の誘導部分グラフを探索できるように拡張する。

3. Tree-Width=2 のグラフを入力としていた部分を、任意の Tree-Width を持つグラフを入力とできるように拡張する。

全域部分グラフと任意の部分グラフの違いは、前者はすべての頂点を使った部分グラフであるの対し、後者は一部の頂点だけを使った部分グラフであることである。よって、拡張の 2 を実現するには、グラフのそれぞれの頂点について「使うのか、使わないのか」の選択を行えばよいことになる。つまり、場合分けを増やせば良いだけのことである。拡張の 3 も、より一般的なグラフについて対応できるように拡張するだけのことである。

一方、木オートマトンをグラフオートマトンに変更するには新たな発見が必要であった。一般認識として、標準的な木オートマトンは存在するのに対し、標準的なグラフオートマトンは存在しない。個別に行われたグラフオートマトンの研究はいくつか存在するが、どれも標準的とは言い難い。木オートマトンは文字列を受理する有限オートマトンを拡張した物であるのに対し、既存のグラフオートマトンの多くは、セルオートマトンを拡張した物である。種となる実装を活用するためには、木オートマトンを拡張してできるグラフオートマトンを新たに発見する必要があった。

2.2 CBG オートマトンの利用

木オートマトンに非常にシンプルな拡張を行うことで、グラフオートマトンが得られることを発見した。この発見は、以下の事実に基づく:

> どんなグラフも、すべてのサイクルを分解すれば、木になる。

グラフから木を構成する様子を図 1 に示す。図のように、すべての閉路から一辺ずつ選択し、それぞれに分解点を二つずつ挿入すれば、木が得られることが分かる。すべてのサイクルを分解して得られる木を、閉路分解グラフ (cycle-broken graph, CBG) と呼ぶことにする。さらに、オートマトンで受理することを容易にするため、それぞれの分解点のペアに接続する二つの辺について、片方は元の辺のラベルそのままとするが、もう片方の辺のラベルは他と区別できるようにアットマーク@とすることにする。また、閉路分解グラフには根となる頂点が指定されており、根付き木となるようにする。

こうして、木オートマトンを拡張し、閉路分解グラフを認識するようなオートマトンを考案することができた。このオートマトンを、**CBG オートマトン**と呼ぶことにする。

CBG オートマトンの詳細は、次章で述べる。

2.3 アルゴリズムの概要

アルゴリズムの基本は、グラフ中に存在する CBG オートマトンに受理される部分グラフ (閉路分解グラフ、つまり根付き木である) の一部らしきものを候補として記憶し、それらを結合して大きくして行くことにある。候補を結合していく途中で、重さが最大または最小にはならない候補は捨てる。そして、最終的に構成された候補 (CBG オートマトンに受理される部分グラフ) の中で、重さが最大または最小のものを一つ出力する。

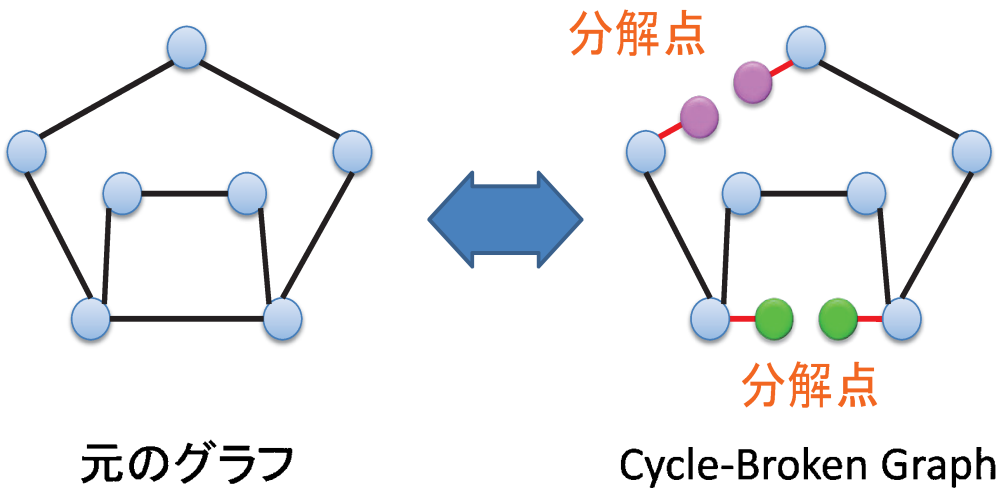


図 1: グラフから木を構成する様子

候補を整理するため、グラフを被覆する複数のハイパーエッジを用意する。各ハイパーエッジは、それが持つ頂点によって誘導される部分グラフ中の必要な候補をすべて保持する。図 2 にハイパーエッジが更新されていく様子を示す。(1) が入力のグラフとする。(2) のように、すべての辺にサイズ 2 のハイパーエッジを用意し、それによってグラフを被覆する。他のハイパーエッジと繋がる頂点をターミナルと呼ぶ。ターミナルを一つずつ消して行くように、ターミナルを共有するハイパーエッジを結合していく。(2) の四隅のターミナルについて、それらを共有するハイパーエッジを結合すると (3) のようになる。さらに、左右中央のターミナルについて、それらを共有するハイパーエッジを結合すると (4) になる。そして、上中央のターミナルについて、それを共有するハイパーエッジを結合すると (5) になる。(5) のハイパーエッジを結合し、最終的には一つのハイパーエッジがグラフ全体を被覆することになる。最後に完成したハイパーエッジが保持する候補が、CBG オートマトンに受理される部分グラフとなる。

候補は、CBG オートマトンに受理される部分グラフであるが、それらは CBG の一部であるため、根付き木の集合となっているはずである。ハイパーエッジを更新する度に、元の候補を結合し、新しい候補を構成する。元の候補の組合せてできる部分グラフの多くは不的確なものとなるため、候補として相応しいものを選別する必要がある。新しい候補の適合性の判定は、次で行うことができる。

1. 結合する各ターミナルにおける状態遷移規則の適応が無矛盾であること。
2. 新しい候補も根付き木の集合となること。

各ターミナルにおける状態遷移規則の適応が無矛盾であるかの判定は、局所的に行えるので簡単である。一方、大域的に、結合後のグラフの形が条件を満たすかどうかの判定は通常難しいが、候補は根付き木の集合なので、形の適合性の判定も非常に容易である。つまり、以下の 2 つだけを確認すればよい。

- 根付き木には、根が正確に 1 つ存在する。

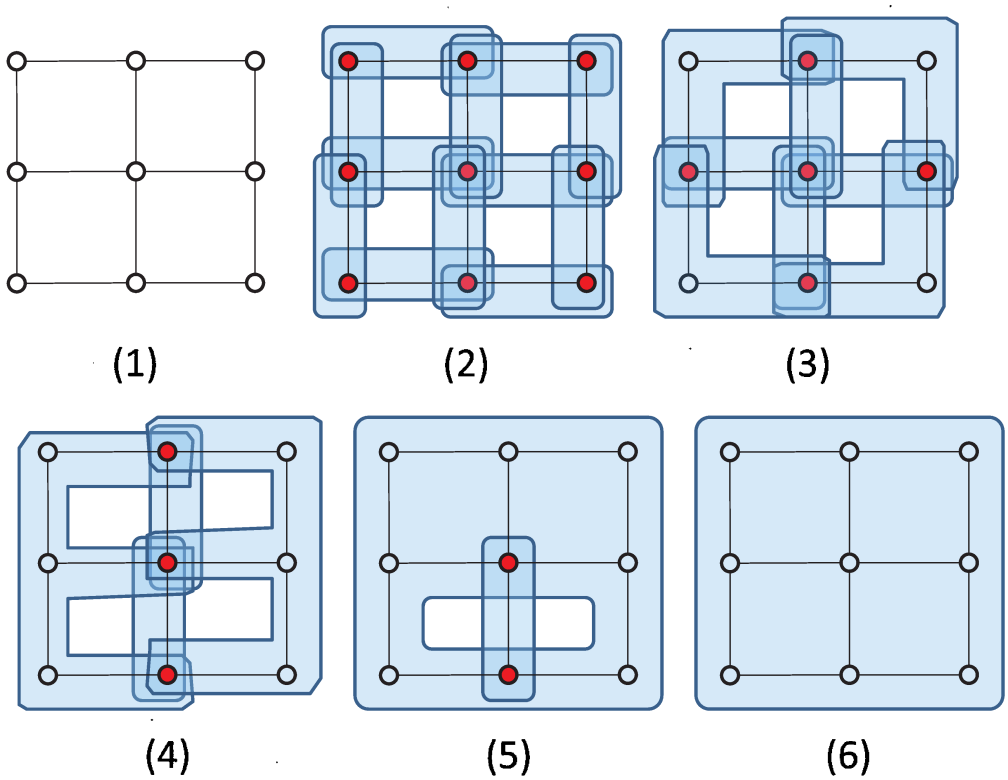


図 2: グラフを被覆する複数のハイパーエッジ

- 根付き木の各頂点の親は、1 つ以下。

Tree-Width が n $(n \geq 2)$ の場合、最大で n 個のターミナルを持つハイパーエッジしか現れない。よって、Tree-Width が小さければ、記憶しておくべき候補の数はそれほど多くはならない。各ターミナルについて、「使うかどうか」、使う場合は「オートマトンが適応する状態遷移規則」、「候補の内部に伝わる状態の組合せ」、「根付き木の根の位置」の別に、最適な候補を 1 個ずつ記憶すればよい。根付き木の根の位置とは、それぞれのターミナルについて、根が「自分自身」、「別のターミナル」、「ターミナル以外の頂点」のいずれであるかを示したものである。

2.4 アルゴリズムの計算量

CBG オートマトンの状態遷移規則の数を $|rules|$、状態遷移規則の幅（右辺の状態数）の最大値を $width$ とする。グラフ G の辺の数を $|E(G)|$、Tree-Width を $tw(G)$ とする。すると、1 つのハイパーエッジが持つ候補の最大数 T は次のようになる。

$$T = (|rules| \cdot 2^{width} \cdot tw(G))^{tw(G)}$$

そして、アルゴリズムの領域量及び時間量は次の式で表される。

$$\text{領域量} = O(|E(G)| \cdot T)$$

$$\text{時間量} = O(|E(G)|^2 \cdot T^2 \cdot \log(T))$$

グラフの最大次数と Tree-Width が定数で押さえられるなら、$width$ と $tw(G)$ も定数で押さえられ、計算量は多項式となる。また、CBG オートマトンのサイズが一定なら、探索して見つかるグラフの大きさは計算量には関係ないことも分かる。

3 CBG オートマトン

3.1 諸定義

グラフとは、順序対 $G = (V, E)$ のことである。ここで、V は**頂点**の有限集合、異なる頂点の組を**辺**と呼び、E は辺の有限集合である。グラフが**連結**であるとは、グラフ上の任意の2頂点間に道が存在することである。**木**とは、連結で閉路の存在しないグラフのことである。ある一つの頂点を選び、それを**根**と呼ぶことにした木を、**根付き木**という。ある頂点に隣接する頂点の内、根とその頂点を結ぶ道上にあるものを、その頂点の**親**という。逆に、ある頂点に隣接する頂点の内、根とその頂点を結ぶ道上にないものを、その頂点の**子**という。根に親はなく、それ以外の頂点には正確に一つ親が存在する。子が一つもない頂点を**葉**という。

Σ を**頂点ラベル**の有限集合、Γ を**辺ラベル**の有限集合とする。グラフ G の**頂点ラベル割り当て**とは，関数 $\sigma : V \to \Sigma$ のことである。グラフ G の**辺ラベル割り当て**とは，関数 $\gamma : E \to \Sigma$ のことである。**ラベル付きグラフ**とは、グラフ (V, E) に、その頂点ラベル割り当てと辺ラベル割り当てをあわせた四つ組 $G = (V, E, \sigma, \gamma)$ のことである。

連結なグラフ $G = (V, E)$ に対し、**閉路分解** $B \subseteq V \times V$ とは、$B' = \{\{u, v\} \mid (u, v) \in B\}$ とすると、$G' = (V, E - B')$ が木となる順序対の集合である。一般にグラフ G の閉路分解は複数存在する。ただし、G が最初から木である場合、$B = \emptyset$ が唯一の閉路分解となる。

ラベル付きグラフ $G = (V, E, \sigma, \gamma)$ と G の閉路分解 B に対し、**閉路分解グラフ** (cycle-broken graph, CBG) とは、次で定義されるラベル付きグラフ $G' = (V', E', \sigma', \gamma')$ のことである。

- $V' = V \cup \{u', v' \mid (u, v) \in B\}$ とする。ここで、u' と v' は V に含まれない新しい頂点である。
- $E' = E - \{\{u, v\} \mid (u, v) \in B\}$
  $+\{\{u, u'\}, \{v, v'\} \mid (u, v) \in B\}$ とする。

頂点ラベル割り当 $\sigma : V' \to \Sigma \cup \{*\}$ と辺ラベル割り当て $\gamma' : E' \to \Gamma \cup \{@\}$ は次のように定義される。

- 各 $v \in V$ について、$\sigma'(v) = \sigma(v)$ とし、各 $(u, v) \in B$ について、$\sigma'(u') = \sigma'(v') = *$ とする。ここで、$*$ は Σ に含まれない特殊な記号である。
- 各 $e \in E$ について、$\gamma'(e) = \gamma(e)$ とし、各 $(u, v) \in B$ に対し、$\gamma(\{u, u'\}) = \gamma(\{u, v\})$、$\gamma(\{v, v'\}) = @$ とする。ここで、$@$ は Γ に含まれない特殊な記号である。

閉路分解グラフは木となる。

3.2 CBGオートマトンの定義

CBG **オートマトン**とは、五つ組 $A = (Q, \Sigma, \Gamma, q_0, R)$ であり、それぞれの要素は次のように定義される。

- Q は**状態**の有限集合である。
- Σ は**頂点ラベル**の有限集合である。
- Γ を**辺ラベル**の有限集合である。
- $q_0 \in Q$ は**開始状態**である。
- R は次の形の**状態遷移規則**の有限集合である。

$$q(f(c_1, c_2, \ldots, c_n)) \rightarrow f(q_1(c_1), q_2(c_2), \ldots, q_n(c_n))$$

ここで、$n \geq 0$, $f \in \Sigma$, $q, q_1, q_2, \cdots, q_n \in Q$, $c_1, c_2, \ldots, c_n \in \Gamma \cup \{@\}$ である。

$G = (V, E, \sigma, \gamma)$ をラベル付きグラフとする。B を G のある閉路分解とし、G の B に対する閉路分解グラフを $G' = (V', E', \sigma', \gamma')$ とする。CBG オートマトン A による閉路分解グラフ G' の**状態割り当て**とは、関数 $S : V' \rightarrow Q$ のことである。$r \in V'$ を G' のある頂点とする。r を根とする G' の状態割り当て S が、以下の条件を満たすとき、A に**受理される**という。

- 根には開始状態が割り当てられる。つまり、$S(r) = q_0$ となる。
- 葉以外の各頂点 v に正確に n 個の子 $v_1, v_2, \ldots, v_n$ があり、$\sigma(v) = f$, $S(v) = q$, $\gamma(\{v, v_1\}) = c_1$, $\gamma(\{v, v_2\}) = c_2$, ..., $\gamma(\{v, v_n\}) = c_n$, $S(v_1) = q_1$, $S(v_2) = q_2$, ..., $S(v_n) = q_n$ となるならば、R は次の状態遷移規則を含む。

$$q(f(c_1, c_2, \ldots, c_n)) \rightarrow f(q_1(c_1), q_2(c_2), \ldots, q_n(c_n))$$

- 各葉 v について、$\sigma(v) = f$, $f \neq *$, $S(v) = q$ となるならば、R は次の状態遷移規則を含む。

$$q(f) \rightarrow f$$

- 各 $(u, v) \in B$ について、$S(u) = S(v)$ となる。

ラベル付きグラフ G が CBG オートマトン A に**受理される**とは、G の閉路分解 B、閉路分解グラフ G'、G' の状態割り当て S 及び G' の頂点 r が存在し、r を根とする G' の状態割り当て S が A に受理されることである。

3.3 CBG オートマトンの能力

CBG オートマトンによって、任意の単純グラフの有限集合を定義することができる。また、パスや木構造の繰り返しがあるグラフの無限集合を定義することができる。

しかし、閉路が存在する部分構造の繰り返しがあるグラフの無限集合は定義できない。状態数よりも多くの閉路が存在する場合、異なる分解点のペアに同じ状態を割り当てなくてはならなくなるためである。

3.4 CBG オートマトンの例

例 1 最初の例は、任意のパスを受理する CBG オートマトンである。このオートマトンは次の 2 つの状態遷移規則で定義される。

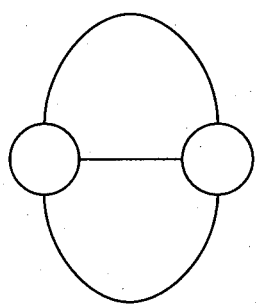
- $q_0(?(-)) \rightarrow ?(q_0(-))$
- $q_0(?) \rightarrow ?$

ここで、? は任意の頂点ラベルにマッチする特殊記号であり、– は任意の辺ラベルにマッチする特殊記号である。パスは閉路を持たないので、これは木オートマトンであるとも言える。

例 2 次の例は、任意の閉路を受理する CBG オートマトンである。このオートマトンは次の 2 つの状態遷移規則で定義される。

- $q_0(?(-, @)) \rightarrow ?(q_1(-), q_1(@))$
- $q_1(?(-)) \rightarrow ?(q_1(-))$

例 3 もう一つの例は、下図のグラフの細分を受理する CBG オートマトンである。このオートマトンは次の 4 つの状態遷移規則で定義される。



- $q_0(?(-, @, @)) \rightarrow ?(q_1(-), q_2(@), q_2(@))$
- $q_1(?(-)) \rightarrow ?(q_1(-))$
- $q_1(?(-, -)) \rightarrow ?(q_2(-), q_2(-))$
- $q_2(?(-)) \rightarrow ?(q_2(-))$

4 実装したプログラムの公開

実装したプログラムは、`http://apricot.cis.ibaraki.ac.jp/CBGfinder/` で公開した。(Google で、「CBGfinder」で検索しても見つかる。) 標準的な C++言語で書かれているので、Windows 上の Visual C++、Linux 上の g++のいずれでもコンパイル可能となっている。

5 まとめと今後の課題

木オートマトンを拡張し、グラフの集合を定義できる CBG オートマトンを考案した。そして、CBG オートマトンに受理される部分グラフを探索する部分グラフ探索アルゴリズムの開発と実装を行った。

今後の課題は、アルゴリズムの領域量を改善することと、もう少し強力なグラフオートマトンを使った部分グラフ探索も行えるようにすることである。

参考文献

[1] Julian R. Ullmann. An algorithm for subgraph isomorphism. *J. ACM*, 23(1):31–42, 1976.

[2] Julian R. Ullmann. Bit-vector algorithms for binary constraint satisfaction and subgraph isomorphism. *ACM Journal of Experimental Algorithmics*, 15, 2010.

[3] Luigi P. Cordella, Pasquale Foggia, Carlo Sansone, and Mario Vento. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(10):1367–1372, 2004.

[4] Jirí Matousek and Robin Thomas. On the complexity of finding iso- and other morphisms for partial k-trees. *Discrete Mathematics*, 108(1-3):343–364, 1992.

[5] 藤芳明生.. ラベル選択付き最小全域木問題と木オートマトンの全域木認識の同時解決と数式 OCR への応用. 2013 年度冬の LA シンポジウム.

[6] Akio Fujiyoshi. A practical algorithm for the uniform membership problem of labeled multidigraphs of tree-width 2 for spanning tree automata. In *Proceedings of 21st International Conference of Implementation and Application of Automata (CIAA 2016), Seoul*, pages 101–112, 2016.